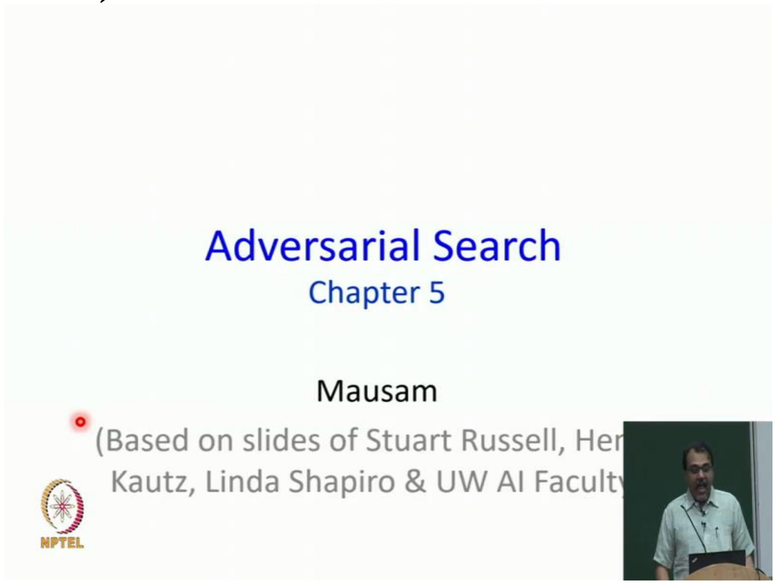
Artificial Intelligence
Prof. Mausam
Department of Computer Science and Engineering
Indian Institute of Technology Delhi

Lecture-29
Minimax Algorithm for Two Player Games-Part-1

So today we are going to talk about adversarial search. And before I start, I do want to mention that as always, you know, these slides have been not only prepared by me, they I am standing on the shoulders of giants and a lot of these slides came from Stuart Russell from UC Berkeley. And then several people actually improved them like Linda Shapiro and other faculty members at University of Washington AI so on so. And I am also using some slides from Henry kautz, who was an ex triple AI president, really famous researcher in the field of AI.

And he was the person who actually taught me the first AI course a lot of what I do today is intended to Henry Kautz, because he sort of got me excited about the field of AI. So, what do we want to do? So, we have so, we have done search algorithms some of them and I have repeatedly mentioned that the search algorithm that you are considering a single agent and atomic search formulation problems and we are going to relax one of those what are we going to relax you are going to relax a single agent assumption here. We are going to play 2 player games in this class now.

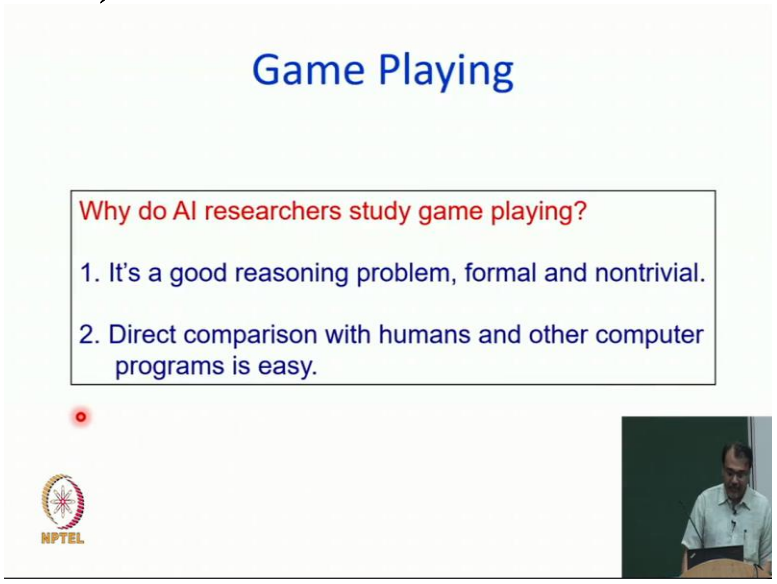
(Refer Slide Time: 01:35)



Adversarial Search
Chapter 5

Mausam
(Based on slides of Stuart Russell, Her
Kautz, Linda Shapiro & UW AI Facult

(Refer Slide Time: 01:45)



Game Playing

Why do AI researchers study game playing?

1. It's a good reasoning problem, formal and nontrivial.
2. Direct comparison with humans and other computer programs is easy.

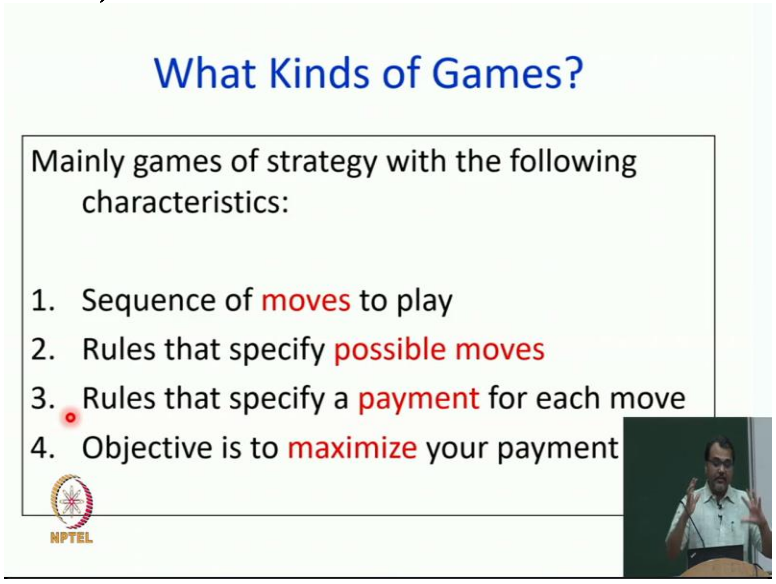
And games have been considered one of the foremost demonstrations of artificial intelligence. Now, why do AI researchers study game playing I mean it is sort of like to have problems if these are like, you know, just fun things, playing chess or playing whatever game tic tac toe whatever. Why do AI researchers engage with these games? So first of all these games are actually even for. I mean, for problem itself, these are good reasoning problems. They are formal problems. So they are easy to specify the rules are well laid out.

It is not like a robot is interacting in a physical environment where lots of things can go wrong, suddenly, it can rain suddenly, it can be sunny, you know, suddenly somebody can come and punch the robot. That is a lot of complexity to deal with in one AI system. On the other hand, if you are thinking about chess or any such game, like a Othello whatever the rules of the game are fakes, you know, you have 16 pieces on each side, and you know, they are very specific ways in which they can execute and so on so.

So, these are formal problems, these are problems which can be easily laid out in a formal abstract specification. And these are not easy problems. I mean, in fact, people humans also think that if you are playing a game at a high quality, then you must be an intelligent person. So therefore, these are good reasoning problems and moreover, they allow direct comparison with humans.

So suppose I create a good AI system which is able to play this game, I can directly compete against humans. And in fact, that becomes my demonstration that my AI has achieved, you know, high quality of intelligence or superhuman performance and things like that. It is harder to do that in the context of let us say, robot control, and so on so forth. But it is easier to do it in the context of a game because we can play against we can defeat we can lose, we can draw and things like that.

(Refer Slide Time: 03:26)



What Kinds of Games?

Mainly games of strategy with the following characteristics:

1. Sequence of moves to play
2. Rules that specify possible moves
3. Rules that specify a payment for each move
4. Objective is to maximize your payment

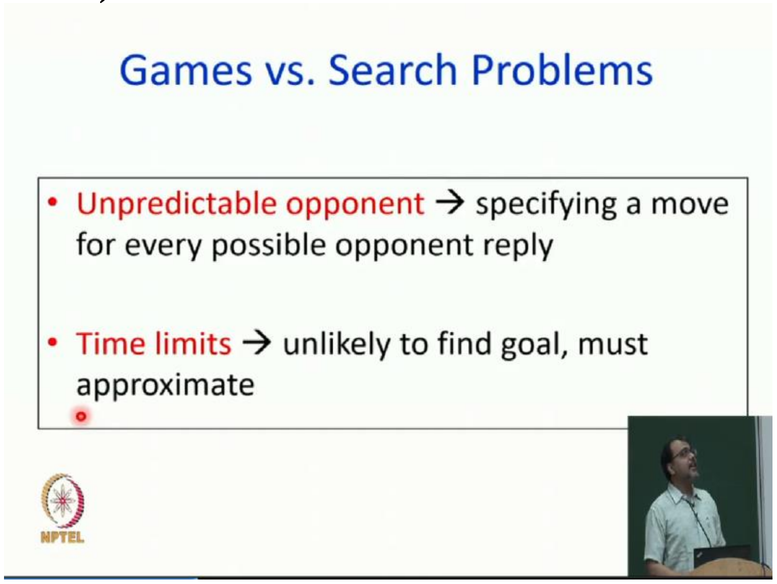
So, what are the different kinds of games? What are the characteristics of the games first of all, we all play moves. We make a move, we change pawn and move it one step forward and so on so forth. And then our adversary, the person we are playing against our opponent also makes a move. Of course, these moves can be done simultaneously. There are some games where there is no notion of you know, I make a move you make a move. Can you think of one such game where there is no sequentially between adversaries racing.

Sure, racing is one such game, where there is no notion of but in the racing really if you think about it, our goal is to do as best as we can. I mean, there may be some strategy where we will come in front of the other person and not let them pass and things like that for like marathon kinds of games, but at least for the 400 meters, 800 meters and so on, there is a relatively straightforward thing that I want to just do as best as I can.

Rock paper scissors, rock paper, scissors is one of the games where I am playing against one adversary. But I am not making one movement than they are making their move. In fact, if you do that in rock, paper, scissors, I will always lose. So, then in addition to the sequence of moves, I will also have rules that specify what are the moves that are allowed and what moves are not allowed in upon can only go 1 or 2 steps depending upon you know, whether it is a first move or later move, also, if the opponent has to attack then it can only attack obliquely so on, so forth.

Now, the moves can bring you money or, you know, reward. And in the game of chess, there is no such reward per say, except if it is a time to change that I am losing some time. But the reward comes at the very end, based on whether I win lose or etc. But there may be other games where each move gives me something back, and our objective will always be to optimize our reward or our payment. So this is how we are going to formulate the various kinds of games.

(Refer Slide Time: 05:50)



Games vs. Search Problems

- Unpredictable opponent → specifying a move for every possible opponent reply
- Time limits → unlikely to find goal, must approximate

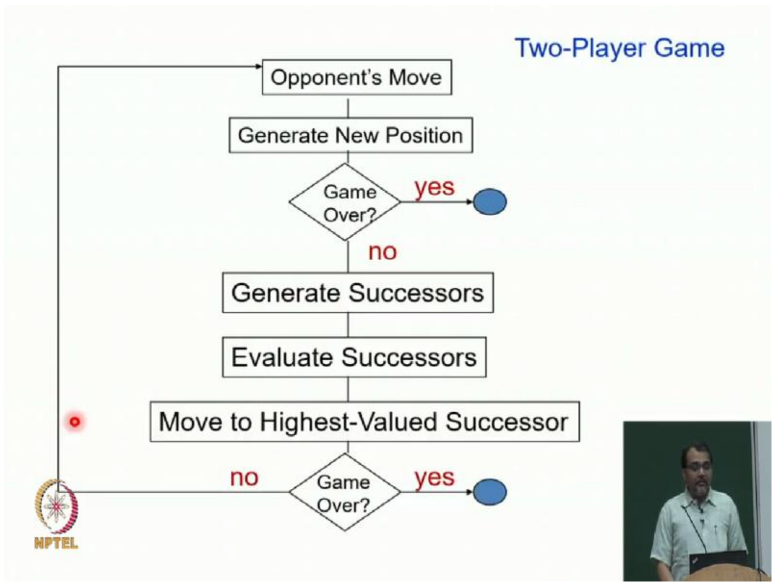
And this is one fundamental difference in which between what we have been studying so far, which are the various kinds of single agent search algorithms, and an adversarial search algorithm, because in single agent search, we have sort of made the assumption that when I take an action, I kind of know exactly where I reach. And later, I am going to take the next action from the state that I reached. That is why I can talk about sequence of steps because all those steps are in my control.

I have a goal to the reach, I can do this action, I can, you know, change, have the gap, go right, have the gap go left or whatever it is in the end puzzle problem. But basically, whatever moves I do, I can exactly know which state I am going to reach and I know that I am going to taking the next move from that particular state. Now that completely changes in gameplay. Why? Because I take an action, now it is not in my control anymore. I do not know what is going to happen next, because my adversary will be making their move.

And they can make any move and I have to be prepared to deal with whatever moves they make. And also I should take the action that reduces their chances of winning or makes me win. But I cannot think of it as a sequence of steps because I mean, not easily we will. But at least at the first at the one set, it does not feel like a simple sequence of steps kind of a problem, because after every action that control is lost, and I do not know exactly what is going to happen. And moreover, in most problems, there are some kind time limits.

So, we may not be able to reach till the very end when we are doing searching. So in depth first search, breadth first search and so on, so forth, we have sort of said that we can reach the goal and we will always output a solution to the goal. But now, we will not be able to do this as you will see in the lecture what will happen is that we will not have the time to search till the end of the game and the game can take 80 160 200 300 place. I make a move, opponent makes the move lots of these and searching till this large depth is going to be extremely difficult. And so, it is not going to be usually possible to find the goal.

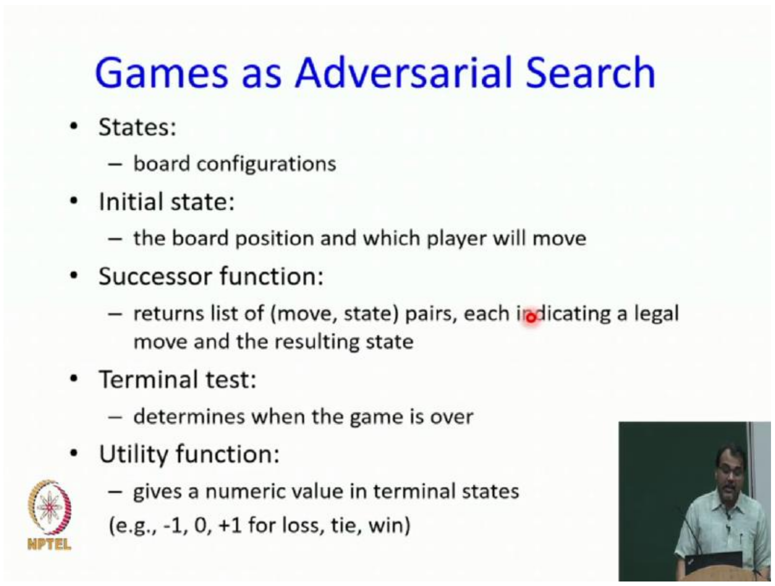
(Refer Slide Time: 08:17)



When you are doing the searching so, we will have to be approximating all the time. We are going to be talking about 2 player sequential games where you know opponent's makes a move it creates a new board position or game position, we check whether the opponent has won or lost if so, great or if not, then we generate successes exactly like we used to do in atomic agent. So, we have now from this board position, we will have many different possible actions.

We will figure out for all these actions what are the different states we can reach and then, we will do something we will evaluate the successes. And most of the magic will be how do I evaluate the successes and then I will be taking the action which seems like the best successor best action and then I will repeat the process. So, this is how my 2 player sequential game is (())(09:01). And of course, as all of AI is search, we have discussed this a little bit a lot of people think that AI search we have to take the search view for this problem for now.

(Refer Slide Time: 09:12)



Games as Adversarial Search

- States:
  - board configurations
- Initial state:
  - the board position and which player will move
- Successor function:
  - returns list of (move, state) pairs, each indicating a legal move and the resulting state
- Terminal test:
  - determines when the game is over
- Utility function:
  - gives a numeric value in terminal states

  (e.g., -1, 0, +1 for loss, tie, win)

And to us games will be some version of search, but we are going to call it adversarial search because as I said, this is not exactly find me the sequence of steps, it is more like I take an action and now the controller is lost. And so for any kind of search algorithm, I need to have a state and initial state a terminal test a successor function and that is not very hard to figure out like we will have board configurations as states suppose.

We have like a board game all the bits of different kind of game than that configuration would be the state, the initial state would be the initial play starting point of the game. A successor function would be what are all the actions that I or the adversary can take indicating all the legal moves and the resulting states? Terminal test would be when the game is over and also whether I won the adversary won or there was a draw and there will be some kind of a utility function.

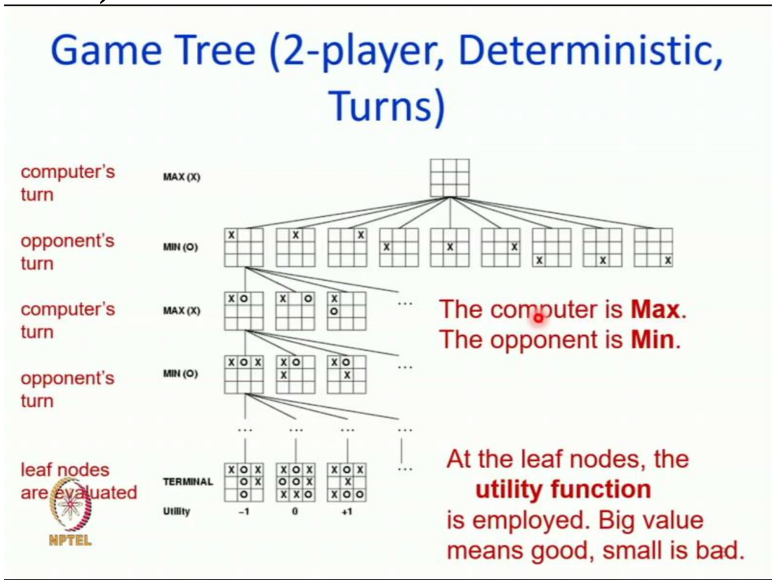
And as a start, we will say that utility function will be minus 1 0 plus 1 and minus 1 would indicate a loss 0 would indicate a draw and plus 1 would indicate that I won. Everybody with all of this .So, now, let us think about this. So, first of all we are trying to maximize my utility function, I want to get to plus 1 for example, if I am trying to maximize my utility function, what is the opponent trying to do? You can always say that it is trying to maximize its utility function.

But in this case in this specific case, it is the utility function is what its utility function? With respect to my utility function, what is my adversary’s utility function? It is negative when I win he or she, the adversary loses when I lose the adversary wins. Let us say the adversary is human and I am the computer because we are thinking from the computers point of view. So I want myself to win which basically also means I want the human to lose.

And similarly the human wants the machine to lose and themselves to win. So, we could have defined 2 different utility functions and thought of a search algorithm which, you know, maximizes my utility in my turn and their utility in their turn. But I am going to simplify this and I am going to only think from the computers point of view. But even then it is not that difficult the problem because at any time I am making a move I want to maximize my utility function.

And anytime the human is taking the move, the human wants to maximize the negative of my utility function which is equal to saying minimize my utility function. So, you can think of this search problem in a very interesting way, where the states are ordered as max nodes and min nodes and max nodes and min nodes.

(Refer Slide Time: 12:27)



Game Tree (2-player, Deterministic, Turns)

The computer is **Max**.
The opponent is **Min**.

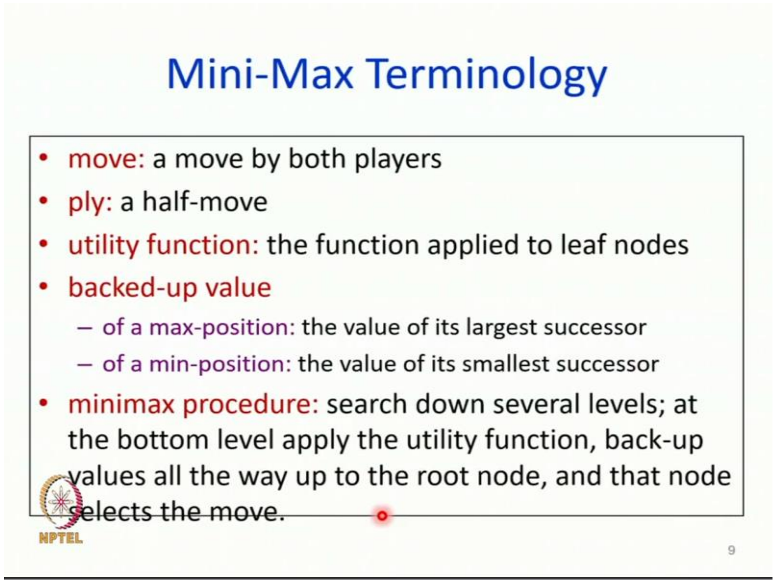
At the leaf nodes, the **utility function** is employed. Big value means good, small is bad.

So, let us think about a search tree variation. So, search tree formulation. Let us say we will look at a simple problem of tic tac toe, the computer starts the computer wants to put its cross and naught its cross such that it maximizes its utility function which is it wins. And then as it is thinking about what is going to happen next, it is say it is going to say that the opponent is going to make their move. And their move would be such that my utility function is minimized. They do not want me to maximize my function they want me to minimize my utility function.

And so this will be like a min node and then I will get my turn and I will be a max node and then they will get a turn and they will be the min node and the game will keep on going eventually we will reach the terminal state, a terminal state in which there is a row like this where there are 3 naughts in a row and here there is a draw and so on. And so if there are 3 naughts in a row I have last somewhere they will be minus 1.

In this case it will be 0 in this case it will be plus 1 and so on so. And this search tree believe it or not has a name and it is called the mini max search tree. Not surprising. I am alternating min and max.

(Refer Slide Time: 13:54)



Mini-Max Terminology

- move: a move by both players
- ply: a half-move
- utility function: the function applied to leaf nodes
- backed-up value
  - of a max-position: the value of its largest successor
  - of a min-position: the value of its smallest successor
- minimax procedure: search down several levels; at the bottom level apply the utility function, back-up values all the way up to the root node, and that node selects the move.

NPTEL

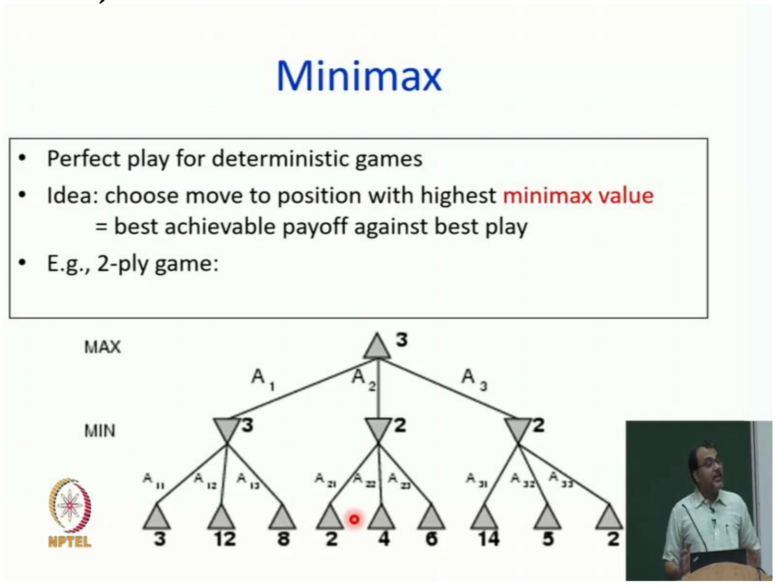
9

So, in mini max parlance, for whatever reason, we will not worry about it in this parlance only in the game playing a move has a special meaning it is not I made a move no, I take my action and that is called ply. And then my opponent takes their action and that is called ply. And together, it is called a move for whatever reason, so we will just learn this and then a utility function would be the function applied at the leaves.

And then we will be backing it up. And I will explain what backing it up means. See backing it up means that if I played this action, in the limit of the game finishing what is the best or the worst utility this node is going to get? That will be the notion of backing up from the leaf back, see because notice that at every step, we are not getting anything. The final value the final utility, we are only getting at the end of the game, and we need to figure out what is the utility of taking my action now.

So I need to back up that value back it up such that I know now what is the utility function of what is the utility of taking this particular action? And a mini max procedure will search down levels at the bottom level, apply the utility function backup all the way back to the root node, and the node will select the move. And in your mind, try to connect it with one of the depth first search breadth first search kinds of algorithms and see we will talk about which algorithm it is.

(Refer Slide Time: 15:29)



Minimax

- Perfect play for deterministic games
- Idea: choose move to position with highest minimax value
  = best achievable payoff against best play
- E.g., 2-ply game:

So by the way, in this slides, I am going to say that triangle sitting upright is a max node and a triangle sitting vertically, the mirror image, opposite is a min node. So this is a max node at the top and this is the min node. And let us say it is a 3 step game. So or 2 step game, I can take an action A 1, A 2, A 3, my opponent can take actions, these different 9 actions depending upon which action I took. And then at the end I get some utility and the utility values are shown at the bottom.

So now if I take action A 1 what should my opponent do? Should it take A 11 A 12 or A13 so it is a damn question. If I decided to take A 11 what should my opponent do? A 11 why? Because my opponent is minimizing and I will after they take A 11 I will get 3 if they take A 12, I will get 12 if they take A 13, I will get 8 and they want to minimize my utility function. So they will always take A 11 and so what is my utility? Best utility best? Not best? What is seeing?

So, there is one thing I do want to say before we move forward. You know, opponents can be terrible. Like, if you are playing with a 3 year old and you are playing tic tac toe. You know, you can defeat them sometimes. But if you are playing a computer which knows its stuff, can you defeat the computer and tic tac toe. No the optimal play is a draw. For now, in this class, we are going to study the game between 2 optimal adversaries. I am as smart as I can be, and my opponent is as smart as they can be.

So when I say what is the utility of A 11, I am going to assume that after I take A 11, my opponent will do the best thing for them, which means the worst thing for me and I will do the best thing for me and they will do the worst thing for me and this game will go on until it reaches the end. So when I asked the question, what is the utility of A 11, I will assume optimal play from there on and in the world of optimal play? What is the utility of A 1? 3.

Because after that, I know that if I took A 1 my opponent is not going to take A 11, and I am going to get 3. Similarly, what is the utility of A 2? So when I take A 2 what will my opponent do A 21 and the utility of A 2 would be 2. And similarly, if I took A 3, my opposition will take A 23 and the utility of A 23 would be 2. So now what should I do at the top? Any confusion in this? We should do A 1 I should do A 1 Why should I do A 1 because I know that I can force my opponent to a move which gives me 3 as my final utility as supposed to taking A 2 or A 3.

In which case, I can only force an opponent to give me 2 if they are playing smartly, and so I will take the one that maximizes my utility function. So see the meaning of backing up the values. These values are at the leaves, but they are being backed up to intermediate nodes based on whether it is a max node or a min node. At the top, I figured out what is my utility at the top and which action is the best. And in fact, not only do I know the best action, I also know the back action of my adversary and my best action and the adversity is best action up to the very end of the game.